

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э.
Баумана
(национальный исследовательский университет)»**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

**по курсу
«Data Science»**

**Тема: «Прогнозирование конечных свойств новых материалов
(композиционных материалов)»**

Слушатель

Матюнин Александр Александрович

Москва, 2023

Содержание

Содержание.....	2
Введение	3
1. Аналитическая часть	5
1.1. Постановка задачи.....	5
1.2. Описание используемых методов.....	7
1.3. Разведочный анализ данных	14
2. Практическая часть	18
2.1. Предобработка данных	18
2.2. Разработка и обучение модели	19
2.3. Тестирование модели	20
2.4. Написать нейронную сеть, которая будет рекомендовать соотношение «матрица-наполнитель».....	21
2.5. Разработка приложения	23
2.6. Создание удалённого репозитория и загрузка	24

Введение

Композиционные материалы — это материалы, состоящие из двух или более компонентов, нерастворимых друг с другом, с чётко обозначенной границей раздела и сильным взаимодействием по всей зоне контакта. Одним из компонентов композитных материалов является непрерывная фаза, он называется матрица, в которой нерастворимые материалы помещаются в другую природу, называемую арматурой или наполнителем.

Внедрение композиционных материалов обусловлено стремлением использовать их преимущества по сравнению с традиционно используемыми металлами и сплавами. Примеры композита – железобетон (сочетание стали арматуры и камня бетона), древесноволокнистая плита ДВП (сочетание древесной основы – щепы и полимерного связующего).

Базальт - магматическая вулканическая порода. Это самая распространённая порода на поверхности Земли и на других планетах Солнечной системы. Базальты образуются путём затвердевания силикатного магматического расплава. Большая часть базальтов образуется на срединно-океанических хребтах и образует океаническую кору. Активно развивается использование композитных материалов на основе базальта.

Базальтопластик - современный композитный материал на основе базальтовых волокон и органического связующего вещества. В настоящее время базальтопластик успешно конкурирует с металлическими изделиями, превосходя их по коррозионной, щелочной, кислотоустойчивости и некоторым другим свойствам. Целью данной работы является прогнозирование конечных свойств новых материалов на основе базальтопластика (композиционных материалов).

Расширение разнообразия материалов, используемых при проектировании нового композиционного материала, увеличивает необходимость определения свойств нового композита при минимальных финансовых затратах. Для решения этой проблемы обычно используются два

способа: физические тесты образцов материалов или оценка свойств, в том числе на основе физико-математических моделей. Традиционно разработка композитных материалов является долгосрочным процессом, так как из свойств отдельных компонентов невозможно рассчитать конечные свойства композита. Для достижения определенных характеристик требуется большое количество различных комбинированных тестов, что делает насущной задачу прогнозирования успешного решения, снижающего затраты на разработку новых материалов и затраты на рабочую силу. Суть прогнозирования заключается в моделировании репрезентативного элемента композитного объема на основе данных о свойствах входящих компонентов (связующего и армирующего компонента). В процессе исследовательской работы были разработаны несколько моделей, способные с высокой вероятностью прогнозировать модули упругости при растяжении и прочности при растяжении, а также были созданы несколько моделей нейронных сетей, которые предлагают соотношение «матрицы - наполнитель». И было создано пользовательское веб - приложение на фреймворке Flask.

Расчет соотношения матрица-наполнитель

Введите параметры

Введите Плотность, кг/м3

Введите Модуль упругости, ГПа

Введите Количество отвердителя, м.%

Введите Содержание эпоксидных групп.%_2

Введите Температура вспышки, C_2

Введите Поверхностная плотность, г/м2

Введите Модуль упругости при растяжении, ГПа

Введите Прочность при растяжении, МПа

Введите Потребление смолы, г/м2

Введите Угол нашивки. град

Введите Шаг нашивки

Введите Плотность нашивки

Рассчитать

Сбросить

Вернуться на главную страницу

Рисунок 1 - скриншот работающего приложения на фреймворке Flask

1. Аналитическая часть

1.1. Постановка задачи

Для исследовательской работы были даны 2 файла: X_br.xlsx (с данными о параметрах базальтопластика, состоящий из 1024 строки и 11 столбцов) и X_nup.xlsx (данными нашивок углепластика, состоящий из 1041 строки и 4 столбцов).

```
4 открыть книгу (покажется панель) и скопировать код в ячейку
df_br = pd.read_excel(r"C:\Users\Александр\Desktop\WORK\KOPPO2171\Data\X_br.xlsx", index_col = "Unnamed: 0")
df_nup = pd.read_excel(r"C:\Users\Александр\Desktop\WORK\KOPPO2171\Data\X_nup.xlsx", index_col = "Unnamed: 0")

# вывести количество строк и данных (столбцов)
print(f"Количество строк в датасете X_br.xlsx: {df_br.shape[0]}")
print(f"Количество строк в датасете X_nup.xlsx: {df_nup.shape[0]}")

Количество строк в датасете X_br.xlsx: 1025
Количество строк в датасете X_nup.xlsx: 1041

# вывести количество колонок в датасете (столбцов)
print(f"Количество колонок в датасете X_br.xlsx: {df_br.shape[1]}")
print(f"Количество колонок в датасете X_nup.xlsx: {df_nup.shape[1]}")

Количество колонок в датасете X_br.xlsx: 10
Количество колонок в датасете X_nup.xlsx: 4

Смотрим данные в табличном виде

df_br.head()
```

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, кг%	Содержание эпоксидных групп, %2	Температура всплытия, С2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2
0	1/05/7143	2030.0	738.736042	30.00	22.267807	100.000000	210.0	70.0	3000.0	220.0
1	1/05/7143	2030.0	738.736042	30.00	23.750000	204.615385	210.0	70.0	3000.0	220.0
2	1/05/7143	2030.0	738.736042	45.50	33.000000	284.615385	210.0	70.0	3000.0	220.0
3	1/05/7143	2030.0	738.736042	125.00	21.250000	300.000000	210.0	70.0	3000.0	220.0
4	2/771331	2030.0	733.000000	111.06	23.267807	284.615385	210.0	70.0	3000.0	220.0

Рисунок 2 – пример начала работы с файлом X_br.xlsx

Цель работы разработать модели для прогноза модуля упругости при растяжении, прочности при растяжении и соотношения «матрица-наполнитель». Для этого нужно объединить 2 файла. Часть информации (17 строк таблицы способов компоновки композитов) не имеют соответствующих строк в таблице соотношений и свойств используемых компонентов композитов, поэтому были удалены.

```
df_nup.head()
```

	Угол нашивки, град	Шаг нашивки	Плотность нашивки
0	0	4.0	57.0
1	0	4.0	60.0
2	0	4.0	70.0
3	0	5.0	47.0
4	0	5.0	57.0

Рисунок 3 - пример начала работы с файлом X_nup.xlsx

Затем провести разведочный анализ данных, нарисовать гистограммы распределения каждой из переменных, диаграммы ящика с усами, попарные графики рассеяния точек.

Диаграммы "ящики с усами" для определения выбросов

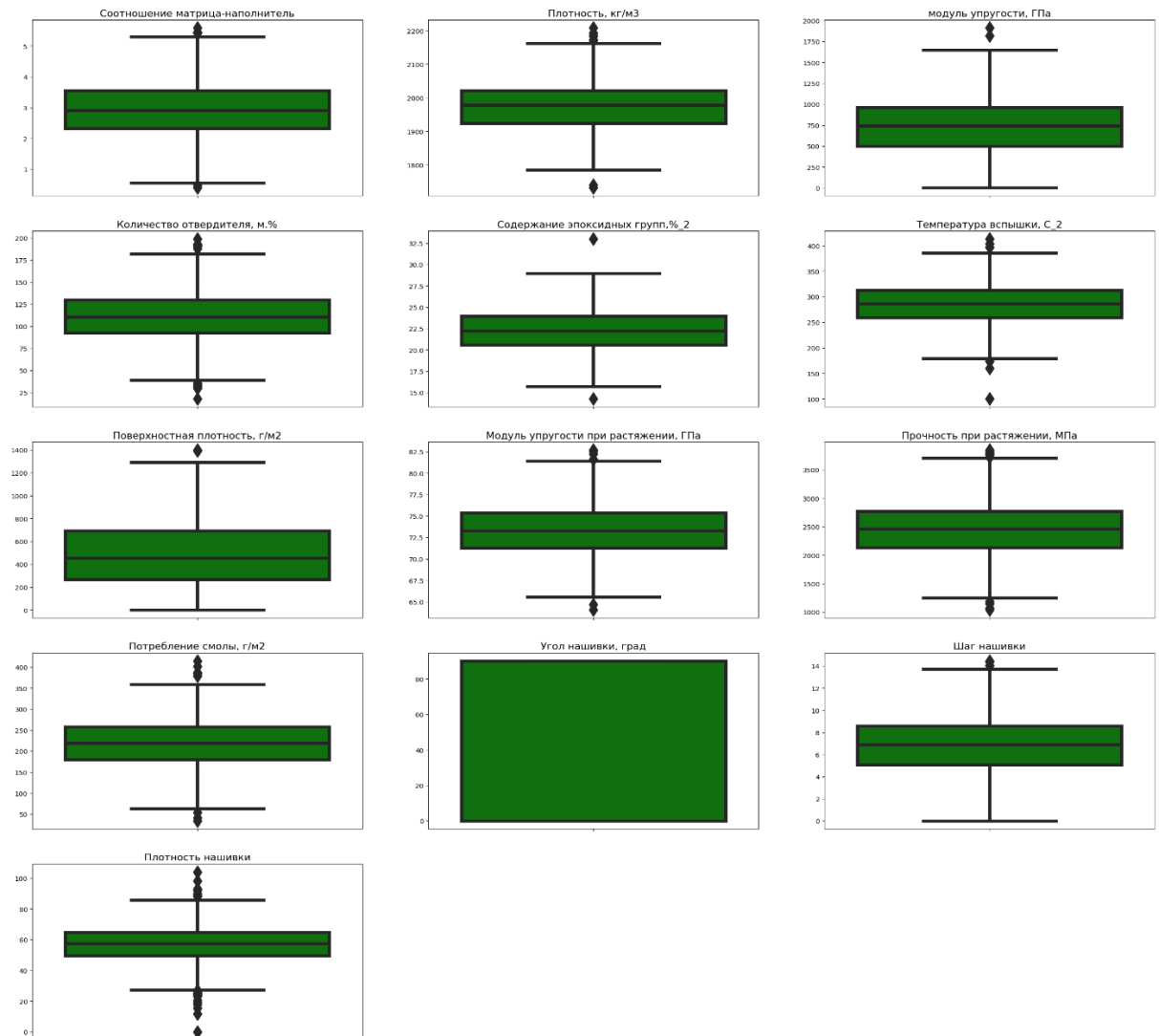


Рисунок 4 - диаграмма "ящик с усами" в объединённом датасете.

Для каждой колонки получить среднее, медианное значение, провести анализ и исключение выбросов, проверить наличие пропусков; пред обработать данные: удалить шумы и выбросы, сделать нормализацию и стандартизацию. Обучить несколько моделей для прогноза модуля упругости при растяжении и прочности при растяжении. Написать нейронную сеть, которая будет рекомендовать соотношение матрица-наполнитель. Разработать приложение с графическим интерфейсом, которое будет выдавать прогноз

соотношения «матрица-наполнитель». Оценить точность модели на тренировочном и тестовом датасете. Создать репозиторий в GitHub и разместить код исследования.

1.2. Описание используемых методов

Данная задача в рамках классификации категорий машинного обучения относится к машинному обучению с учителем и традиционно это задача регрессии. Цель любого алгоритма обучения с учителем — определить функцию потерь и минимизировать её, поэтому для наилучшего решения в процессе исследования были применены следующие методы:

- К-ближайших соседей;
- Линейная регрессия;
- Стохастический градиентный спуск
- случайный лес;
- градиентный бустинг;
- модель многослойного перцептрона
- деревья решений;

Метод ближайших соседей - К-ближайших соседей (kNN - k Nearest Neighbours) ищет ближайшие объекты с известными значения целевой переменной и основывается на хранении данных в памяти для сравнения с новыми элементами. Алгоритм находит расстояния между запросом и всеми примерами в данных, выбирая определенное количество примеров (k), наиболее близких к запросу, затем голосует за наиболее часто встречающуюся метку (в случае задачи классификации) или усредняет метки (в случае задачи регрессии).

Достоинства метода: прост в реализации и понимании полученных результатов; имеет низкую чувствительность к выбросам; не требует построения модели; допускает настройку нескольких параметров; позволяет делать дополнительные допущения; универсален; находит лучшее решение из возможных; решает задачи небольшой размерности.

Недостатки метода: замедляется с ростом объёма данных; не создаёт правил; не обобщает предыдущий опыт; основывается на всем массиве доступных исторических данных; невозможно сказать, на каком основании строятся ответы; сложно выбрать близость метрики; имеет высокую зависимость результатов классификации от выбранной метрики; полностью перебирает всю обучающую выборку при распознавании; имеет вычислительную трудоёмкость.

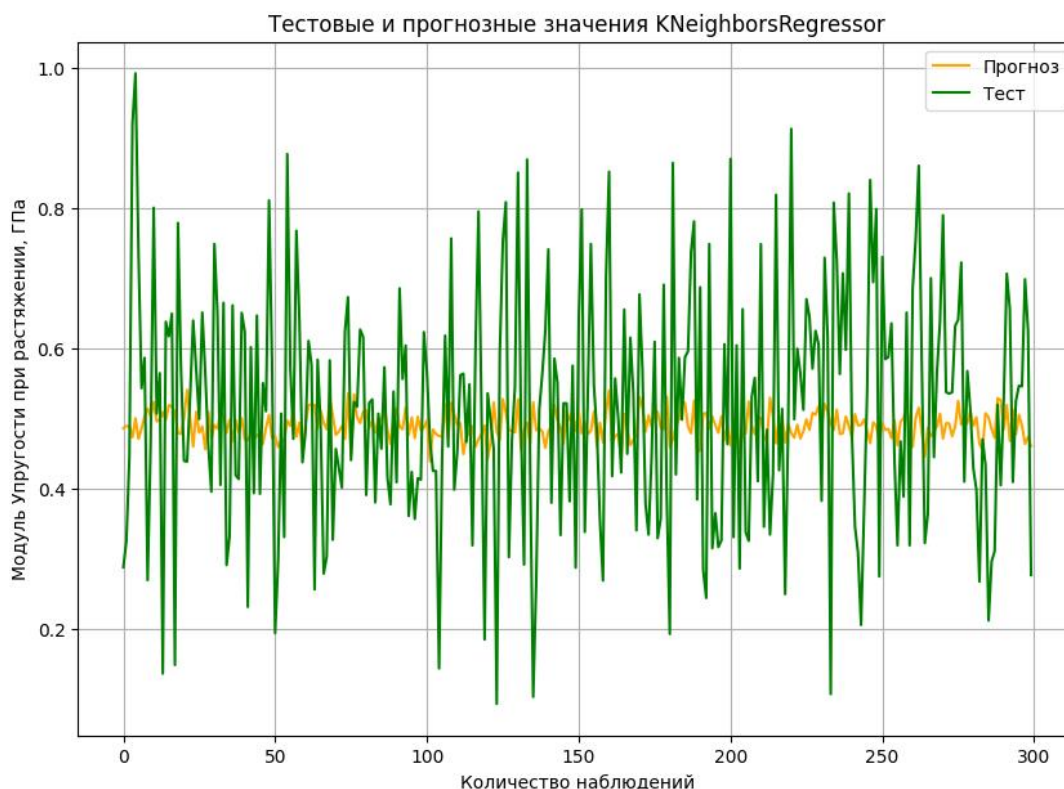


Рисунок 5 - график K-ближайших соседей для упругости при растяжении, ГПа

Линейная регрессия (Linear regression) — это алгоритм машинного обучения, основанный на контролируемом обучении, рассматривающий зависимость между одной входной и выходными переменными. Это один из самых простых и эффективных инструментов статистического моделирования. Она определяет зависимость переменных с помощью линии наилучшего соответствия. Модель регрессии создаёт несколько метрик. R^2 , или коэффициент детерминации, позволяет измерить, насколько модель может объяснить дисперсию данных. Если R-квадрат равен 1, это значит, что

модель описывает все данные. Если же R-квадрат равен 0,5, модель объясняет лишь 50 процентов дисперсии данных. Оставшиеся отклонения не имеют объяснения. Чем ближе R^2 к единице, тем лучше.

Достоинства метода: быстр и прост в реализации; легко интерпретируем; имеет меньшую сложность по сравнению с другими алгоритмами;

Недостатки метода: моделирует только прямые линейные зависимости; требует прямую связь между зависимыми и независимыми переменными; выбросы оказывают огромное влияние, а границы линейны.

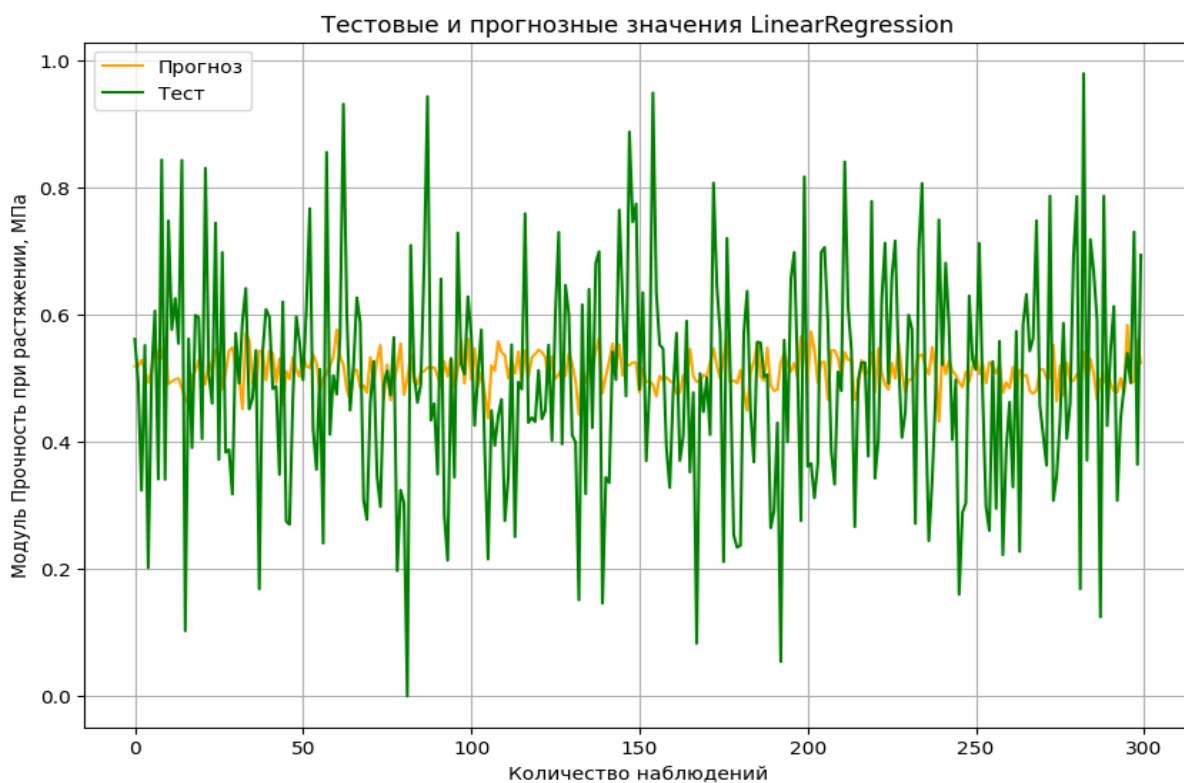


Рисунок 6 - график линейной регрессии для прочности при растяжении, МПа

Случайный лес (RandomForest) — это множество решающих деревьев. Универсальный алгоритм машинного обучения с учителем, представитель ансамблевых методов. Если точность дерева решений оказалась недостаточной, мы можем множество моделей собрать в коллектив.

Достоинства метода: не переобучается; не требует предобработки входных данных; эффективно обрабатывает пропущенные данные, данные с большим числом классов и признаков; имеет высокую точность предсказания

и внутреннюю оценку обобщающей способности модели, а также высокую параллелизуемость и масштабируемость.

Недостатки метода: построение занимает много времени; сложно интерпретируемый; не обладает возможностью экстраполяции; может недообучаться; трудоёмко прогнозируемый; иногда работает хуже, чем линейные методы.

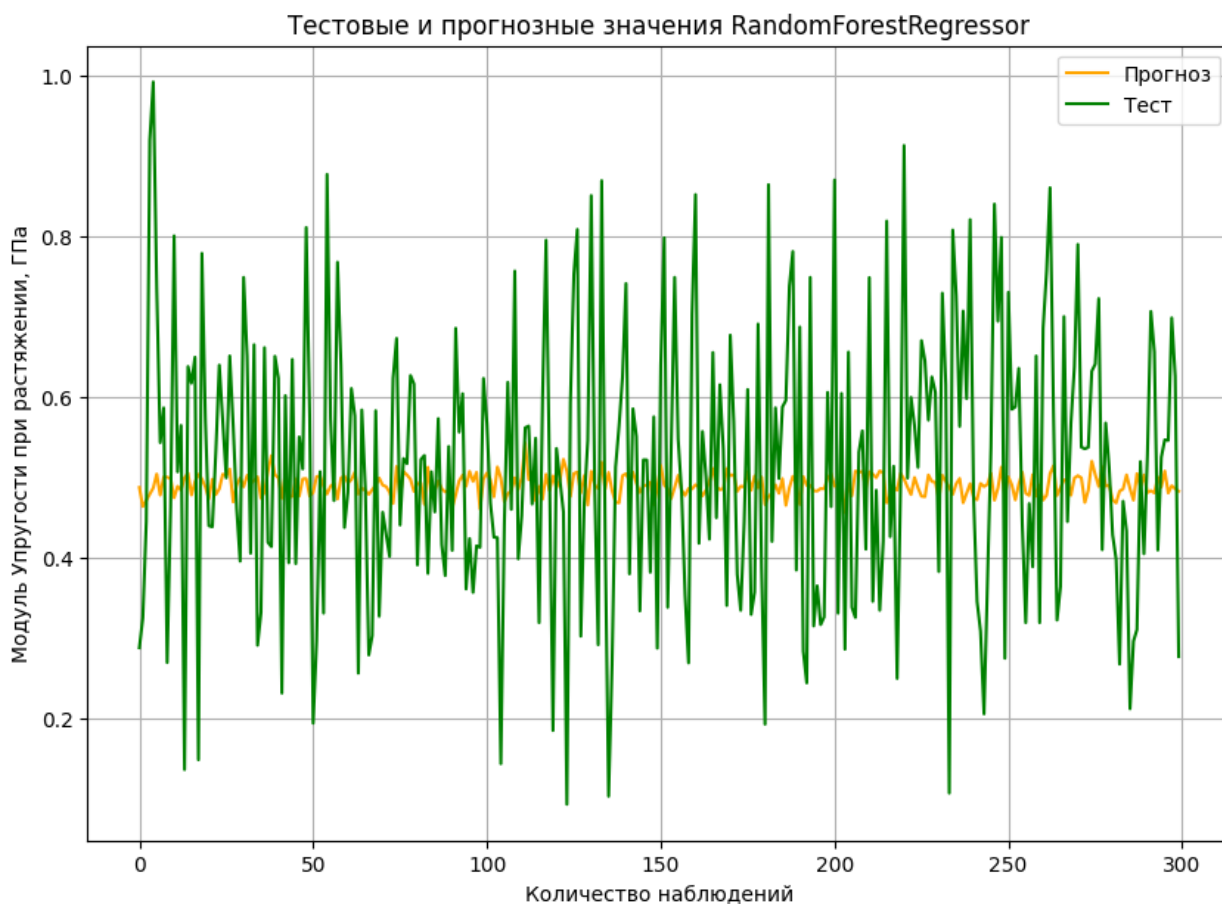


Рисунок 7 - график "случайного леса" для упругости при растяжении, ГПа

Многослойный перцептрон (MLP) – это алгоритм обучения с учителем, который может изучить аппроксиматор нелинейной функции как для классификации так и для регрессии. Класс (MLPRegressor) реализует многослойный перцептрон, который обучается с использованием обратного распространения без функции активации. Следовательно, он использует квадратную ошибку как функцию потерь, а на выходе представляет собой набор непрерывных значений.

MLP тренирует с использованием стохастического градиентного спуска, Адама или L-BFGS. Стохастический градиентный спуск(SGD)

обновляет параметры, используя градиент функции потерь по отношению к параметру, который требует адаптации.

Многослойный персептрон чувствителен к масштабированию функций, поэтому рекомендуется масштабировать данные, что бы имели среднее значение 0 и дисперсию 1.

Преимущества персептрона в том что достаточно прост в использовании, настройки по умолчанию позволяющие быстро оценивать общую производительность нашей модели.

К недостаткам можно отнести что ускорение на GPU или TPU недоступно, нет возможности внести дополнительные слои, не подходит для работы с неструктурированными данными, такие как изображения , звук или текст.

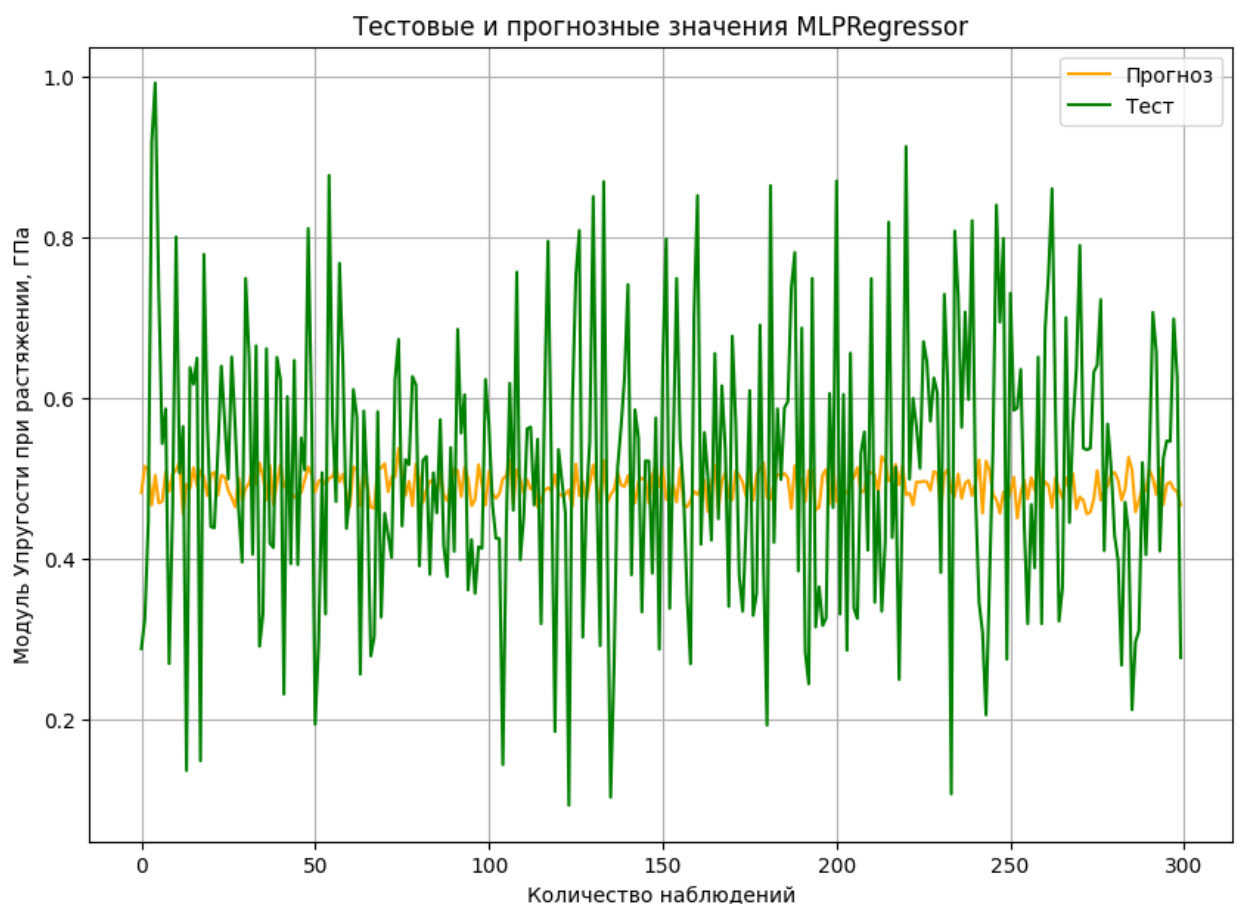


Рисунок 6 - график многослойного персептрона для упругости при растяжении, ГПа

Градиентный бустинг (Gradient Boosting) — это ансамбль деревьев решений, обученный с использованием градиентного бустинга. В основе данного алгоритма лежит итеративное обучение деревьев решений с целью

минимизировать функцию потерь. Основная идея градиентного бустинга: строятся последовательно несколько базовых классификаторов, каждый из которых как можно лучше компенсирует недостатки предыдущих. Финальный классификатор является линейной композицией этих базовых классификаторов.

Достоинства метода: новые алгоритмы учатся на ошибках предыдущих; требуется меньше итераций, чтобы приблизиться к фактическим прогнозам; наблюдения выбираются на основе ошибки; прост в настройке темпа обучения и применения; легко интерпретируем.

Недостатки метода: необходимо тщательно выбирать критерии остановки, иначе это может привести к переобучению; наблюдения с наибольшей ошибкой появляются чаще; слабее и менее гибко чем нейронные сети.

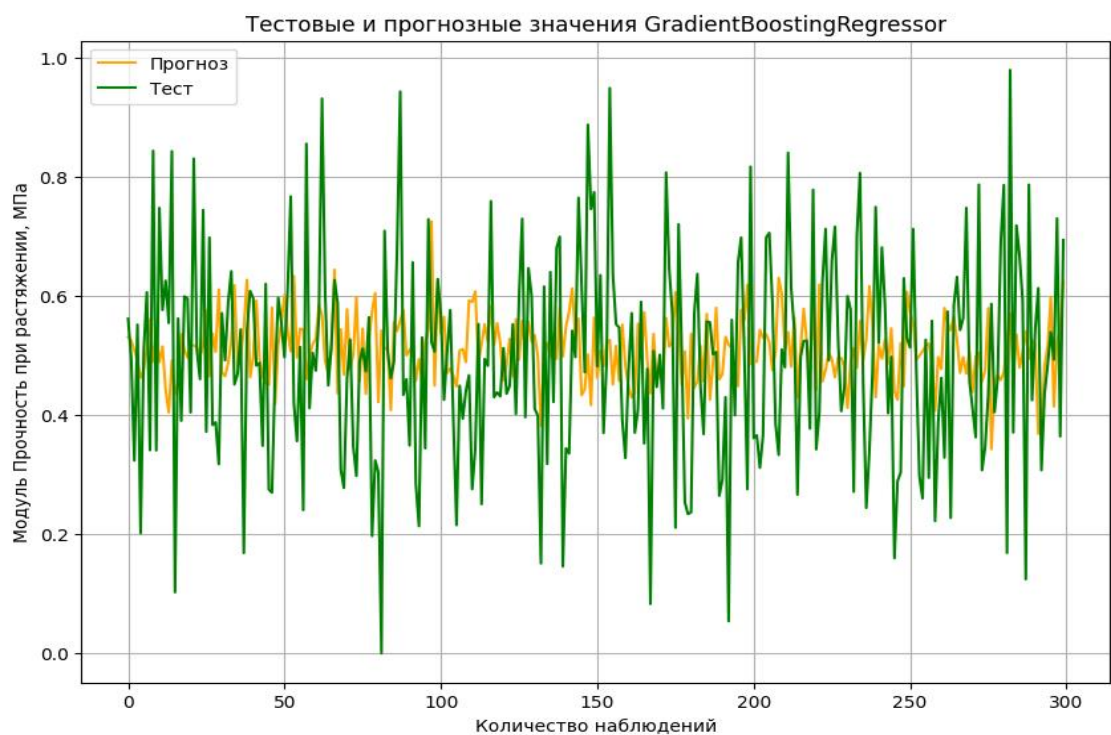


Рисунок 9 - график градиентного бустинга для упругости при растяжении, ГПа

Немного расскажем об используемых метриках качества моделей: R² или коэффициент детерминации измеряет долю дисперсии, объяснённую моделью, в общей дисперсии целевой переменной.

Если он близок к единице, то модель хорошо объясняет данные, если же он близок к нулю, то качество прогноза идентично средней величине целевой переменной (т.е. очень низкое). Отрицательные значения коэффициента детерминации означают плохую объясняющую способность модели.

models_uprygost.T							
Model	KNeighborsRegressor_uprygost	LinearRegression_uprygost	SGDRegressor_uprygost	RandomForestRegressor_uprygost	GradientBoostingRegressor_uprygost	MLPRegressor_uprygost	DecisionTreeRegressor_uprygost
MASE	0.744	0.744	0.749	0.747	0.754	0.746	0.755
MAPE	0.339	0.339	0.349	0.34	0.34	0.34	0.343
MSE	0.029	0.029	0.029	0.029	0.03	0.029	0.03
R2 score	-0.013	-0.013	-0.002	-0.016	-0.035	-0.018	-0.035
Best score	-0.027145	-0.039372	-0.076482	-0.040744	-0.062736	-0.020307	-0.025649

models_prochnost.T							
Model	KNeighborsRegressor_prochnost	LinearRegression_prochnost	SGDRegressor_prochnost	RandomForestRegressor_prochnost	GradientBoostingRegressor_prochnost	MLPRegressor_prochnost	DecisionTreeRegressor_prochnost
MASE	0.689	0.697	0.709	0.692	0.733	0.69	0.701
MAPE	inf	inf	inf	inf	inf	inf	inf
MSE	0.03	0.03	0.03	0.03	0.032	0.03	0.03
R2 score	-0.008	-0.006	-0.037	-0.006	-0.083	-0.013	-0.031
Best score	-0.029803	-0.025615	-0.075167	-0.009232	-0.032024	-0.006573	-0.01405

Рисунок 7 – общая таблица результатов метрик.

MSE (Mean Squared Error) или средняя квадратичная ошибка принимает значения в тех же единицах, что и целевая переменная. Чем ближе к нулю MSE, тем лучше работают предсказательные качества модели.

```

dtr_prochnost_result = pd.DataFrame({
    'Model': 'DecisionTreeRegressor_prochnost',
    'MASE': mase(y_prochnost_test, dtr_prochnost_pred, y_prochnost_train).round(3),
    'MAPE': mape(y_prochnost_test, dtr_prochnost_pred).round(3),
    'MSE': mean_squared_error(y_prochnost_test, dtr_prochnost_pred).round(3),
    'R2 score': r2_score(y_prochnost_test, dtr_prochnost_pred).round(3),
    "Best score": grid_search_dtr_prochnost.best_score_
}, index=['Модуль Прочности при растяжении'])
models_prochnost = pd.concat([models_prochnost, dtr_prochnost_result])

```

Рисунок 11 - код для отображения различных метрик и сохранения данных в датафрейм.

1.3. Разведочный анализ данных

Прежде чем передать данные в работу моделей машинного обучения, необходимо обработать и очистить их. Очевидно, что «грязные» и необработанные данные могут содержать искажения и пропущенные значения – это ненадёжно, поскольку способно привести к крайне неверным результатам по итогам моделирования. Но безосновательно удалять что-либо тоже неправильно. Именно поэтому сначала набор данных надо изучить.

```
# рассчитаем основные статистические показатели
df_nb.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	2.930366	0.913222	0.389403	2.317887	2.906878	3.552660	5.591742
Плотность, кг/м3	1023.0	1975.734888	73.729231	1731.764635	1924.155467	1977.621657	2021.374375	2207.773481
модуль упругости, ГПа	1023.0	739.923233	330.231581	2.436909	500.047452	739.664328	961.812526	1911.536477
Количество отвердителя, м.%	1023.0	110.570769	28.295911	17.740275	92.443497	110.564840	129.730366	198.953207
Содержание эпоксидных групп,%_2	1023.0	22.244390	2.406301	14.254985	20.608034	22.230744	23.961934	33.000000
Температура вспышки, С_2	1023.0	285.882151	40.943260	100.000000	259.066528	285.896812	313.002106	413.273418
Поверхностная плотность, г/м2	1023.0	482.731833	281.314690	0.603740	266.816645	451.864365	693.225017	1399.542362
Модуль упругости при растяжении, ГПа	1023.0	73.328571	3.118983	64.054061	71.245018	73.268805	75.356612	82.682051
Прочность при растяжении, МПа	1023.0	2466.922843	485.628006	1036.856605	2135.850448	2459.524526	2767.193119	3848.436732
Потребление смолы, г/м2	1023.0	218.423144	59.735931	33.803026	179.627520	219.198882	257.481724	414.590628
Угол нашивки, град	1023.0	44.252199	45.015793	0.000000	0.000000	0.000000	90.000000	90.000000
Шаг нашивки	1023.0	6.899222	2.563467	0.000000	5.080033	6.916144	8.586293	14.440522
Плотность нашивки	1023.0	57.153929	12.350969	0.000000	49.799212	57.341920	64.944961	103.988901

Рисунок 12 - описательная статистика датасета

Цель разведочного анализа - получение первоначальных представлений о характерах распределений переменных исходного набора данных, формирование оценки качества исходных данных (наличие пропусков, выбросов), выявление характера взаимосвязи между переменными с целью последующего выдвижения гипотез о наиболее подходящих для решения задачи моделях машинного обучения.

```
# Проверим датасет на дубликаты
df_nb.duplicated().sum()
#Дубликатов нет
```

Рисунок 13 - проверка датасета на наличие дубликатов

В качестве инструментов разведочного анализа используется: оценка статистических характеристик датасета; гистограммы распределения каждой из переменной (несколько различных вариантов); диаграммы ящика с усами (несколько интерактивных вариантов); попарные графики рассеяния точек (несколько вариантов); график «квантиль-квантиль»; тепловая карта (несколько вариантов); описательная статистика для каждой переменной; анализ и исключение выбросов; проверка наличия пропусков и дубликатов.

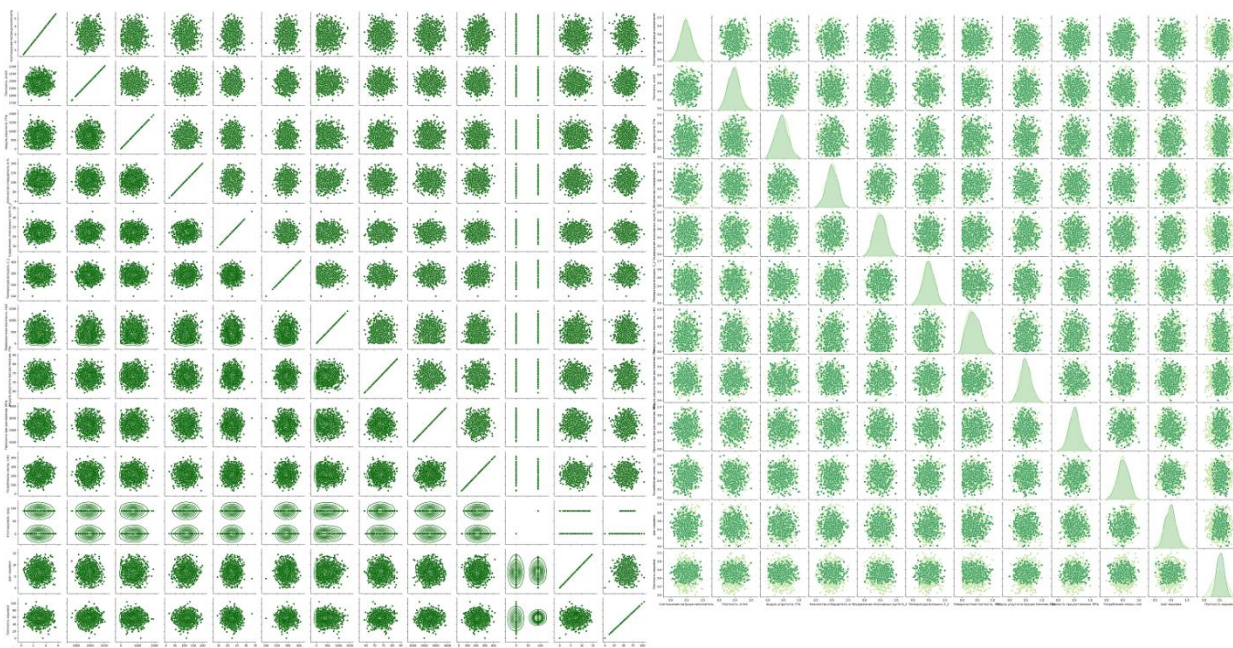


Рисунок 14 - попарные графики рассеяния точек (два разных варианта)

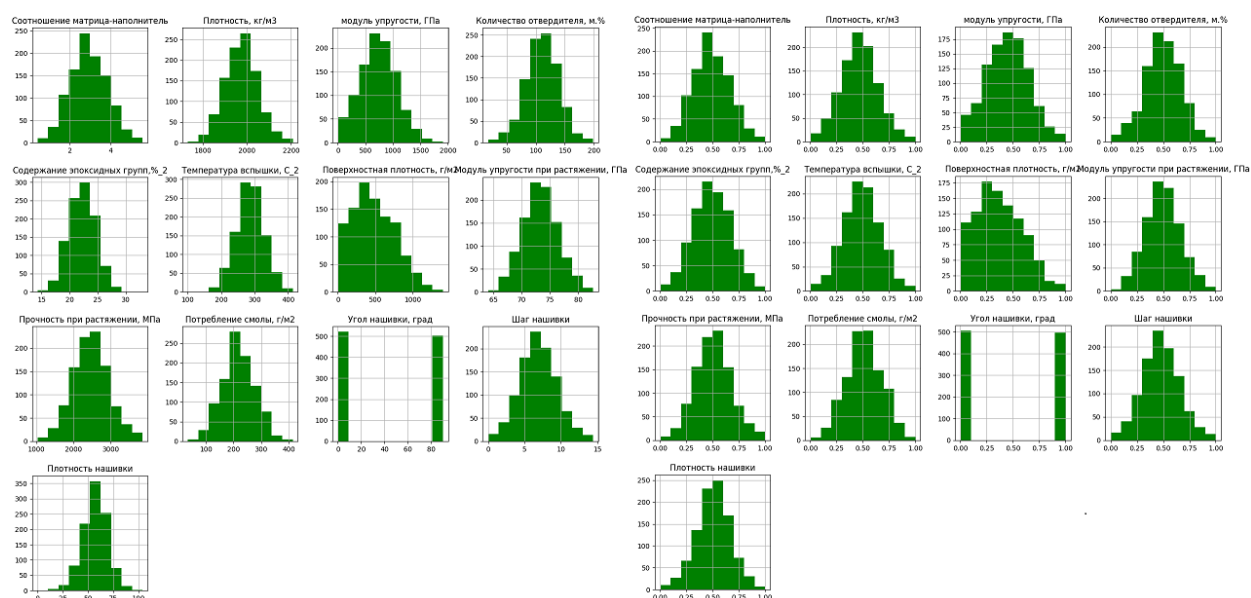


Рисунок 15 - гистограммы распределения (2 разных варианта)

Гистограммы используются для изучения распределений частот значений переменных. Мы видим очень слабую корреляцию между переменными.

График вероятности для распределения данных

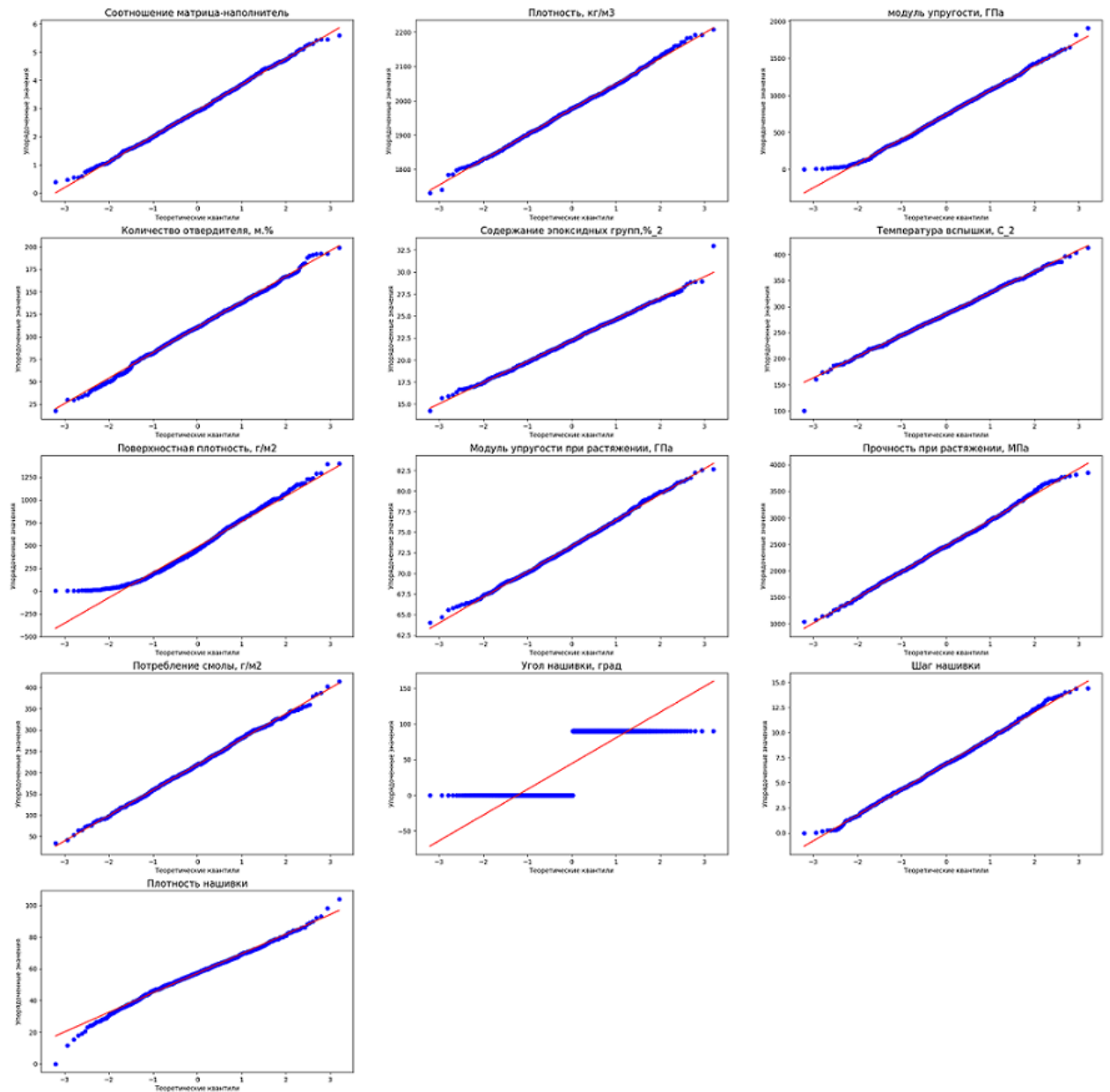


Рисунок 16 - графики «квантиль-квантиль»

Диаграммы "ящики с усами" для определения выбросов

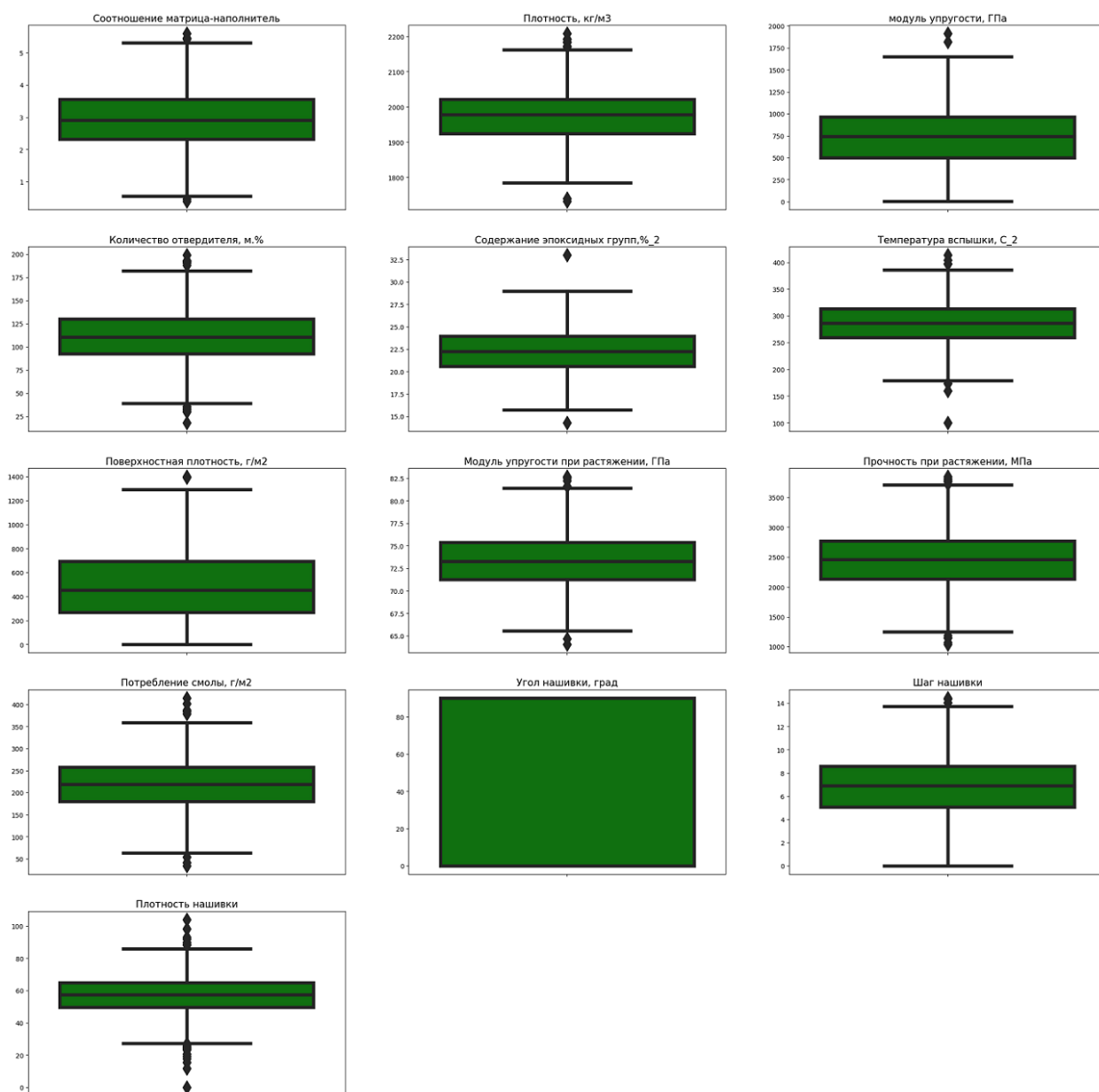


Рисунок 17 - график "ящиков с усами" для всех переменных

После обнаружения выбросов данные, значительно отличающиеся от выборки, будут полностью удалены. Для расчёта этих данных мы будем использовать методы трех сигм и межквартильного расстояния.

Данные объединённого датасета не имеют чётко выраженной зависимости, что подтверждает тепловая карта с матрицей корреляции и матрицы диаграмм рассеяния.

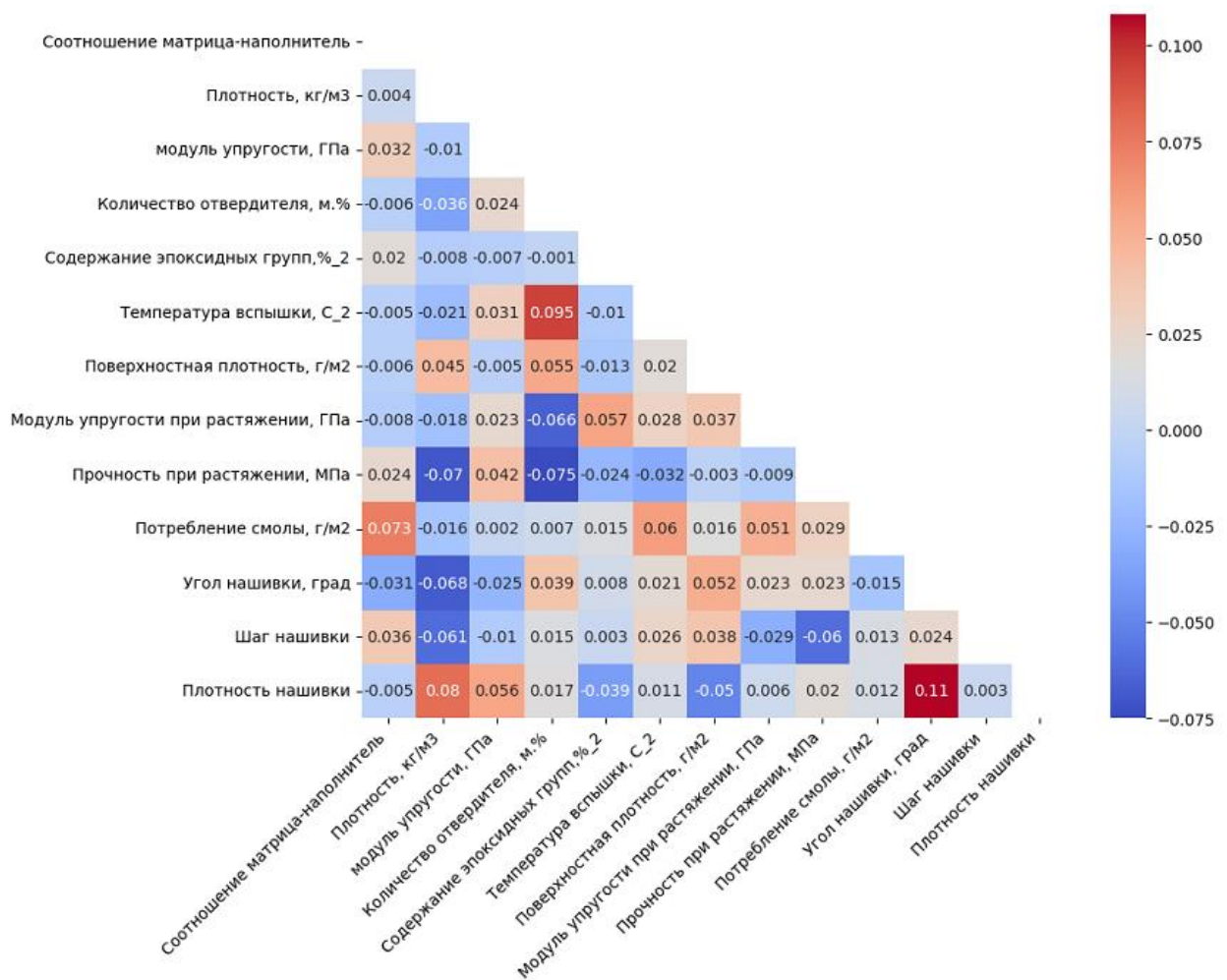


Рисунок 18 - тепловая карта с корреляцией данных.

Максимальная корреляция между плотностью нашивки и углом нашивки 0.11, значит нет зависимости между этими данными. Корреляция между всеми параметрами очень близка к 0, корреляционные связи между переменными не наблюдаются.

2. Практическая часть

2.1. Предобработка данных

В ходе проведённого анализа принимаю решение столбец "Угол нашивки" не приводить к виду «0» и «1».

По условиям задания нормализуем значения. Для этого применим MinMaxScaler.

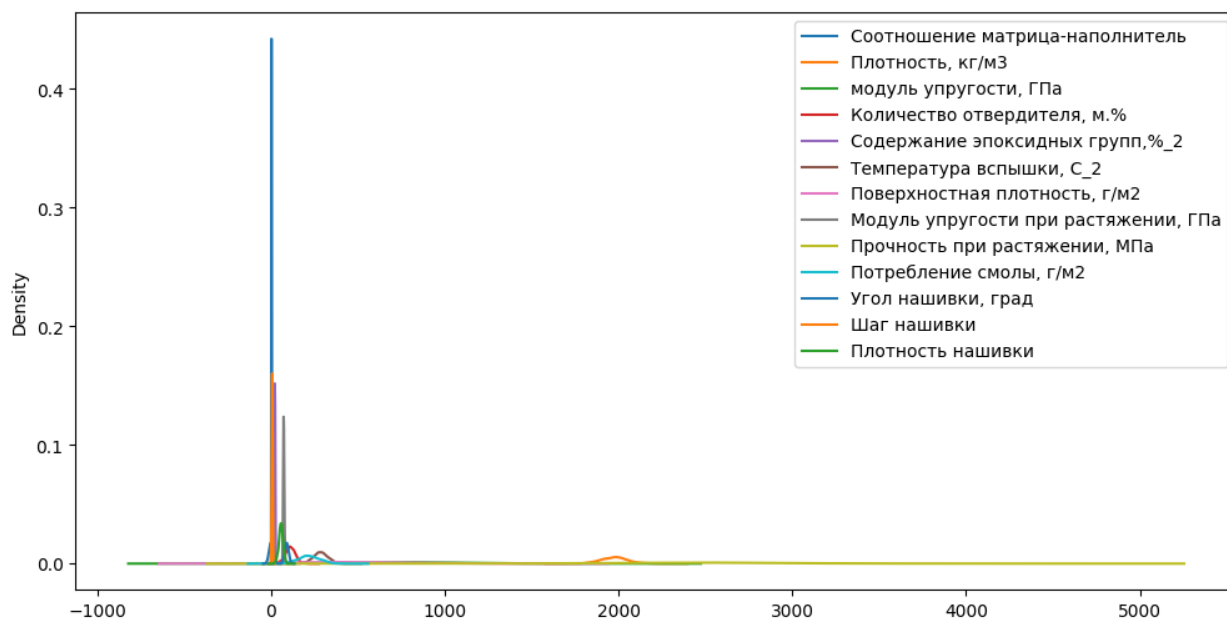


Рисунок 20 - визуализированные данные нормализации.

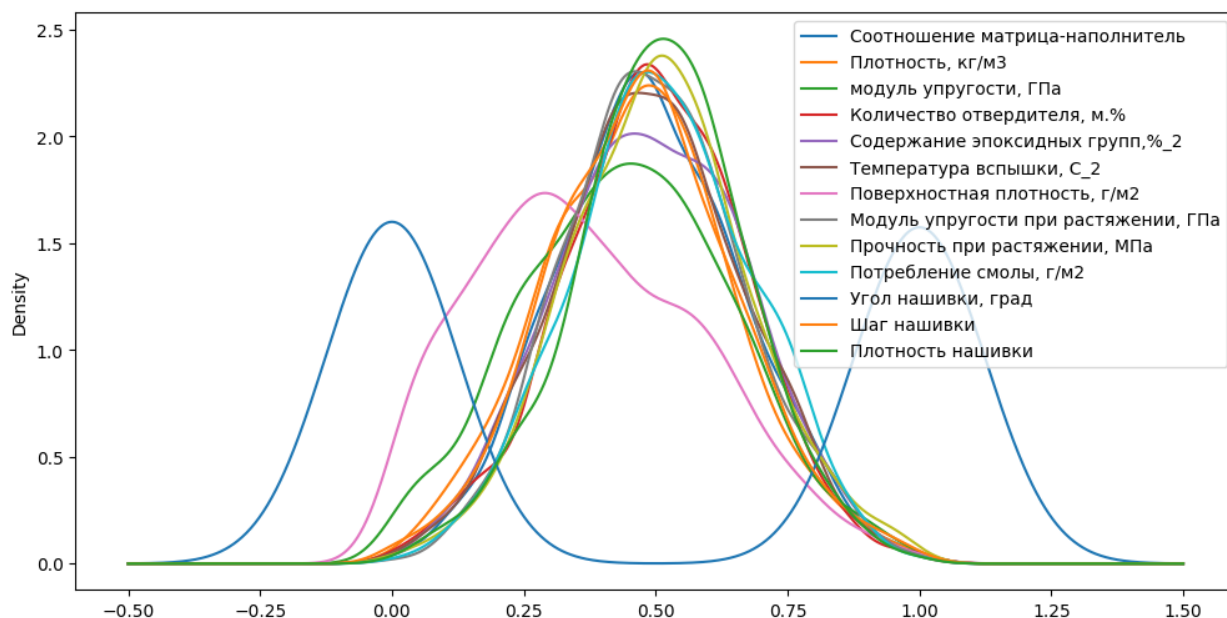


Рисунок 21 - визуализированные данные нормализации.

2.2. Разработка и обучение модели

Разработка и обучение моделей машинного обучения осуществлялась для двух выходных параметров: «Прочность при растяжении» и «Модуль упругости при растяжении» отдельно. Для решения применим все методы, описанные выше.

```

knn_uprygost = KNeighborsRegressor()
param_grid = {
    "n_neighbors": range(1, 301, 2),
    "weights": ['uniform', 'distance'],
    "algorithm": ['auto', 'ball_tree', 'kd_tree', 'brute'],
    "p": range(1, 5, 2)
}
grid_search_knn_uprygost = GridSearchCV(knn_uprygost, param_grid, cv=10, verbose=1, scoring = 'neg_mean_squared_error')
grid_search_knn_uprygost.fit(X_uprygost_train, y_uprygost_train)
model_knn_uprygost = grid_search_knn_uprygost.best_estimator_
model_knn_uprygost.fit(X_uprygost_train, y_uprygost_train)
knn_uprygost_pred = model_knn_uprygost.predict(X_uprygost_test)
knn_uprygost_Bscore = grid_search_knn_uprygost.best_score_
print("Best params", grid_search_knn_uprygost.best_params_)
print("Best score", knn_uprygost_Bscore)

(43)

""" Fitting 10 folds for each of 2400 candidates, totalling 24000 fits
Best params {'algorithm': 'auto', 'n_neighbors': 73, 'p': 3, 'weights': 'uniform'}
Best score -0.02714483206176299

```

Рисунок 22- поиск гиперпараметров и обучение модели

Порядок разработки модели для каждого параметра и для каждого выбранного метода можно разделить на следующие этапы: разделение нормализованных данных на обучающую и тестовую выборки (в соотношении 70 на 30%); проверка моделей при стандартных значениях; сравнение с результатами модели, выдающей среднее значение; создание графика; сравнение моделей по метрике MAE; поиск сетки гиперпараметров, по которым будет происходить оптимизация модели. В качестве параметра оценки выбран коэффициент детерминации (R^2); оптимизация подбора гиперпараметров модели с помощью выбора по сетке и перекрёстной проверки; подстановка оптимальных гиперпараметров в модель и обучение модели на тренировочных данных; оценка полученных данных; сравнение со стандартными значениями.

Прочность при растяжении и модуль упругости не имеет линейной зависимости. Все использованные модели не справились с задачей. Результат неудовлетворительный. Свойства композитных материалов в первую очередь зависят от используемых материалов.

2.3. Тестирование модели

После обучения моделей была проведена оценка точности этих моделей на обучающей и тестовых выборках. В качестве параметра оценки модели

использовалась средняя квадратическая ошибка (MSE). Результат неудовлетворительный.

Хотя в целом при таких результатах можно применять среднее значение переменной в качестве прогнозного.

2.4. Написать нейронную сеть, которая будет рекомендовать соотношение «матрица – наполнитель».

Обучение нейронной сети— это такой процесс, при котором происходит подбор оптимальных параметров модели, с точки зрения минимизации функционала ошибки. Начнём строить нейронную сеть с помощью класса `keras.Sequential`.

```
df_mn = df_min_max['Соотношение матрица-наполнитель']
df_non_mn = df_min_max.drop(['Соотношение матрица-наполнитель'], axis=1)
X_train, X_test, y_train, y_test = train_test_split(df_non_mn, df_mn,
                                                    test_size = 0.3,
                                                    random_state = 23,
                                                    shuffle = True)
```

Рисунок 23 – разделение нормализованного датасета обучающую и проверочную выборки.

```

model_minmax = Sequential()
model_minmax.add(Input(shape=X_train.shape[1]))
model_minmax.add(Dense(128))
model_minmax.add(Dense(128, activation='relu'))
model_minmax.add(Dense(128, activation="ELU"))
model_minmax.add(Dense(128, kernel_regularizer=keras.regularizers.l2(0.01),
                        activation='relu'))
model_minmax.add(Dense(24, activation='ELU'))
model_minmax.add(Dense(12, activation='relu'))
model_minmax.add(Dense(1, activation='linear'))

model_minmax.summary()

```

```

Model: "sequential"

```

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	1664
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 128)	16512
dense_3 (Dense)	(None, 128)	16512
dense_4 (Dense)	(None, 24)	3096
dense_5 (Dense)	(None, 12)	300
dense_6 (Dense)	(None, 1)	13

```

Total params: 54609 (213.32 KB)
Trainable params: 54609 (213.32 KB)
Non-trainable params: 0 (0.00 Byte)

```

Рисунок 24 - построение первой нейросети.

Обучим и оценим модель, посмотрим на потери, зададим функцию для визуализации факт/прогноз для результатов моделей.

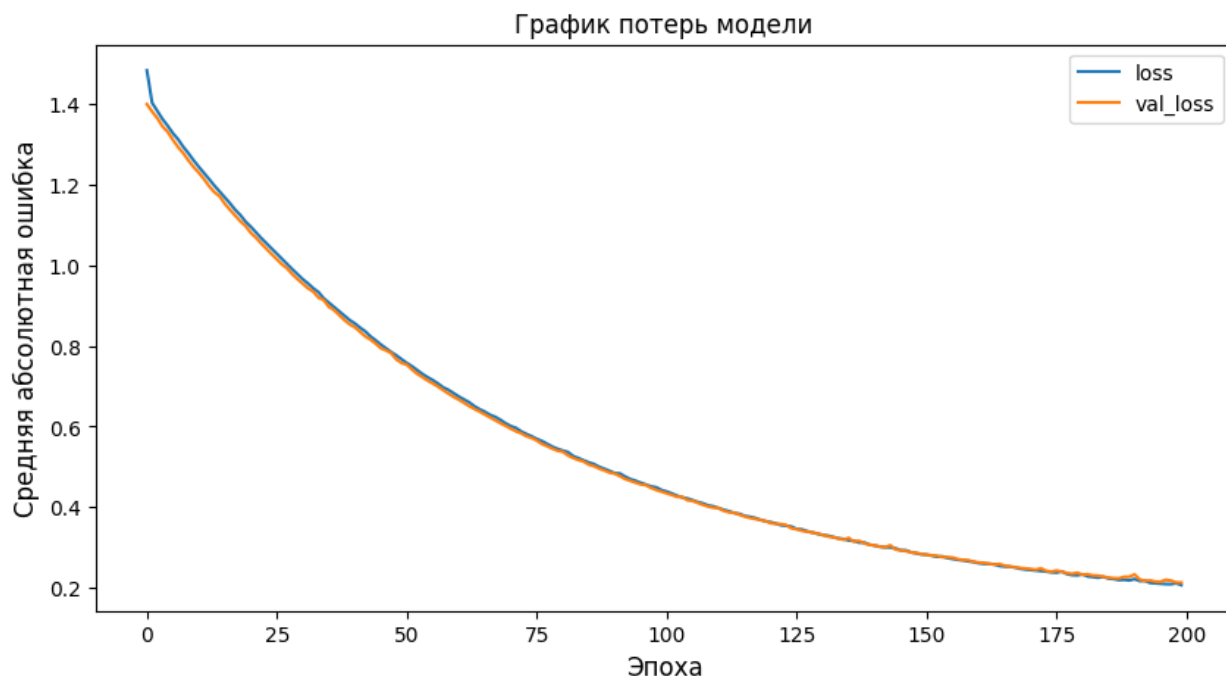


Рисунок 25 - график потерь модели 1



Рисунок 26 - график прогнозных и тестовых значений.

2.5. Разработка приложения

Приложение успешно работает и показывает результат прогноза для «Модуль упругости при растяжении, ГПа»

Расчет соотношения матрица-наполнитель

Введите параметры

Введите Плотность, кг/м³

Введите Модуль упругости, ГПа

Введите Количество отвердителя, м.%

Введите Содержание эпоксидных групп, %₂

Введите Температура вспышки, С₂

Введите Поверхностная плотность, г/м²

Введите Модуль упругости при растяжении, ГПа

Введите Прочность при растяжении, МПа

Введите Потребление смолы, г/м²

Введите Угол нашивки, град

Введите Шаг нашивки

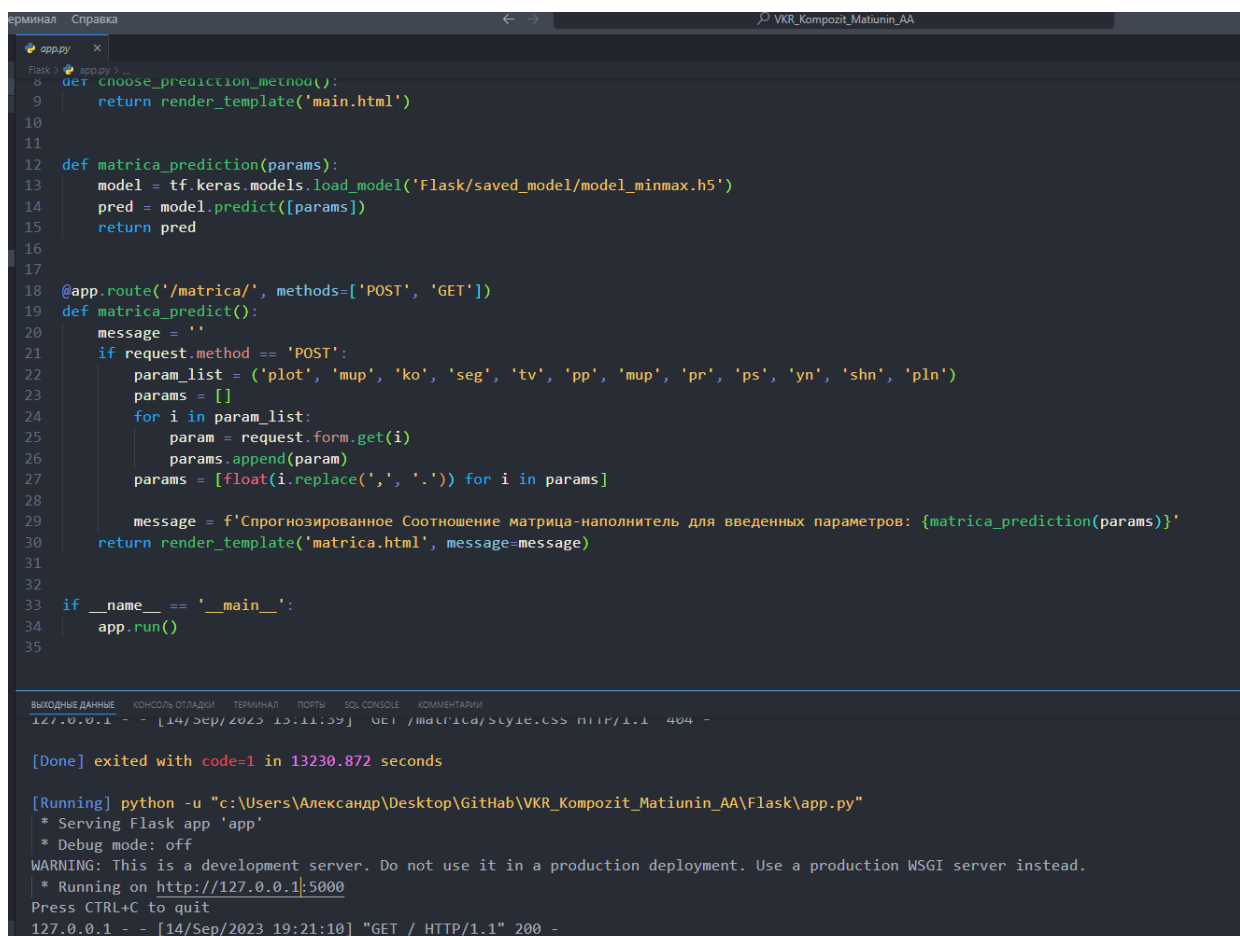
Введите Плотность нашивки

Спрогнозированное Соотношение матрица-наполнитель для введенных параметров: [[14.564138]]

Рисунок 27 - пример результата работы приложения

Данное приложение — это основной файл Flask, папка templates, с шаблоном html - страницы, папка saved_model с сохранённой моделью для

данных, CSS-папка со стилями, для удобства чтения кода. При запуске приложения, пользователь переходит на: <http://127.0.0.1:5000/>.



```
from flask import Flask, request, render_template

app = Flask(__name__)

def choose_prediction_method():
    return render_template('main.html')

def matrica_prediction(params):
    model = tf.keras.models.load_model('Flask/saved_model/model_minmax.h5')
    pred = model.predict([params])
    return pred

@app.route('/matrica/', methods=['POST', 'GET'])
def matrica_predict():
    message = ''
    if request.method == 'POST':
        param_list = ('plot', 'mup', 'ko', 'seg', 'tv', 'pp', 'mup', 'pr', 'ps', 'yn', 'shn', 'pln')
        params = []
        for i in param_list:
            param = request.form.get(i)
            params.append(param)
        params = [float(i.replace(',', '.')) for i in params]

        message = f'Спрогнозированное Соотношение матрица-наполнитель для введенных параметров: {matrica_prediction(params)}'
    return render_template('matrica.html', message=message)

if __name__ == '__main__':
    app.run()
```

ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ SQL CONSOLE КОММЕНТАРИИ

```
127.0.0.1 - - [14/Sep/2023 19:11:39] "GET /matrica/style.css HTTP/1.1" 404 -

[Done] exited with code=1 in 13230.872 seconds

[Running] python -u "c:\Users\Александр\Desktop\GitHab\VKR_Kompozit_Matiunin_AA\Flask\app.py"
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [14/Sep/2023 19:21:10] "GET / HTTP/1.1" 200 -
```

Рисунок 28 - часть кода приложения и ссылка для открытия html - файла

В открывшемся окне пользователю необходимо ввести в соответствующие ячейки требуемые значения и нажать на кнопку «Отправить».

На выходе пользователь получает результат прогноза для значения параметра «Модуль упругости при растяжении, ГПа»

2.6. Создание удалённого репозитория и загрузка

Репозиторий был создан на github.com по адресу: https://github.com/Matiunin1982/VKR_Kompozit_Matiunin_AA

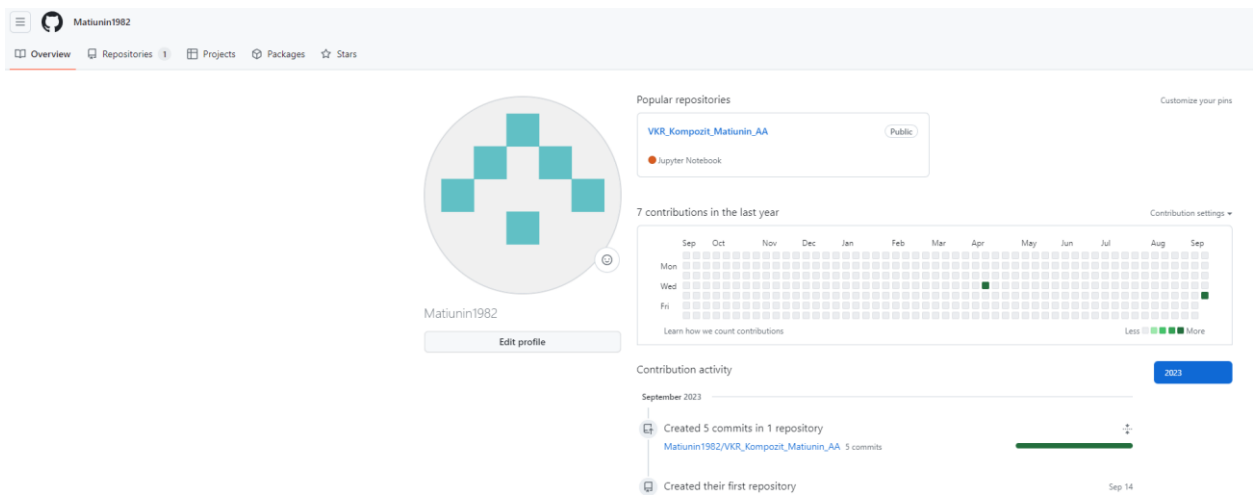


Рисунок 29 - часть страницы на github.com